Bioinformatics

Day 2, 2020-09-01

Xiaofan Zhou

How to change the sources for your Ubuntu

- /etc/apt/sources.list

```
xiaofan@Xiaofan-ThinkPad: ~/practical/2019_Fall_Bioinfo/week_04/data$ more /etc/apt/sources.list
# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted
# deb-src http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted

## Major bug fix updates produced after the final release of the
## distribution.
deb http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted
# deb-src http://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted
```

1. sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak
   备份！备份！！备份！！！
2. sed ‘s/security.ubuntu.com/mirrors.tuna.tsinghua.edu.cn/’ \
   /etc/apt/sources.list.bak | \
   sed ‘s/archive.ubuntu.com/ mirrors.tuna.tsinghua.edu.cn /’ \
   > sources.list
3. sudo cp sources.list /etc/apt/sources.list

A quick review of last week…

- Command line
- Filesystem
- Permissions
- Environment variables

Environment variable

- "*a dynamic-named value that can affect the way running processes will behave on a computer*."

- $PATH

  check: echo $PATH

  set (temporarily): export PATH=$HOME/usr/bin:$PATH

  set (permanently): modify ~/.bashrc

Day 2

- Software installation
- Local BLAST

Nightmares in bioinformatics

- File formats
- Software installation
- Versions, parameters, reference builds…

Three common ways to install a software on Linux

- apt (or apt-get)
  - sudo apt update & apt upgrade
  - sudo apt install ncbi-blast+
- precompiled binaries
  - tar xf diamond-linux64.tar.gz
- compile from source

Compile from source #1

- configure & make
  - tar xf hmmer-3.3.1.tar.gz
  - cd hmmer-3.3.1
  - ./configure (*--prefix=/home/xiaofan/usr/bin/hmmer*)
  - make (*-j 2*)
  - make check
  - make install

Compile from source #2

- cmake & make
  - tar xf diamond-2.0.4.tar.gz
  - cd diamond-2.0.4
  - mkdir build
  - cd build
  - cmake .. (*-DCMAKE_INSTALL_PREFIX=/home/xiaofan/usr/bin*)
  - make (*-j 2*)
  - make install

Perl modules & Python libraries

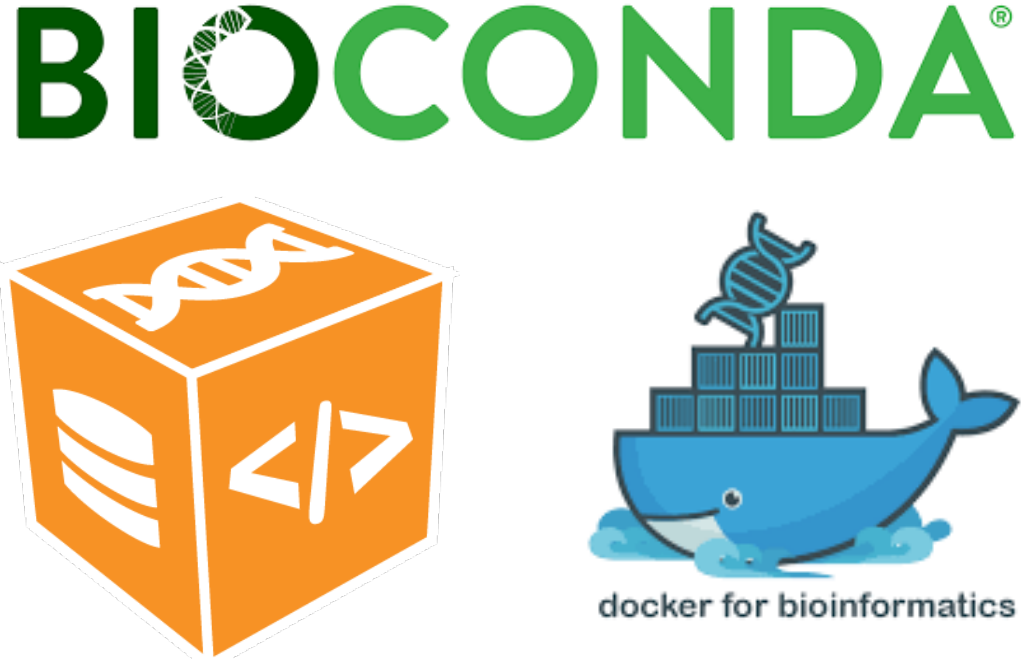
Perl: CPAN

- perl -MCPAN -e shell
- install XXX

Python: pip

- pip install --user XXX
- pip3 install --user XXX

Conda, Docker, Github…

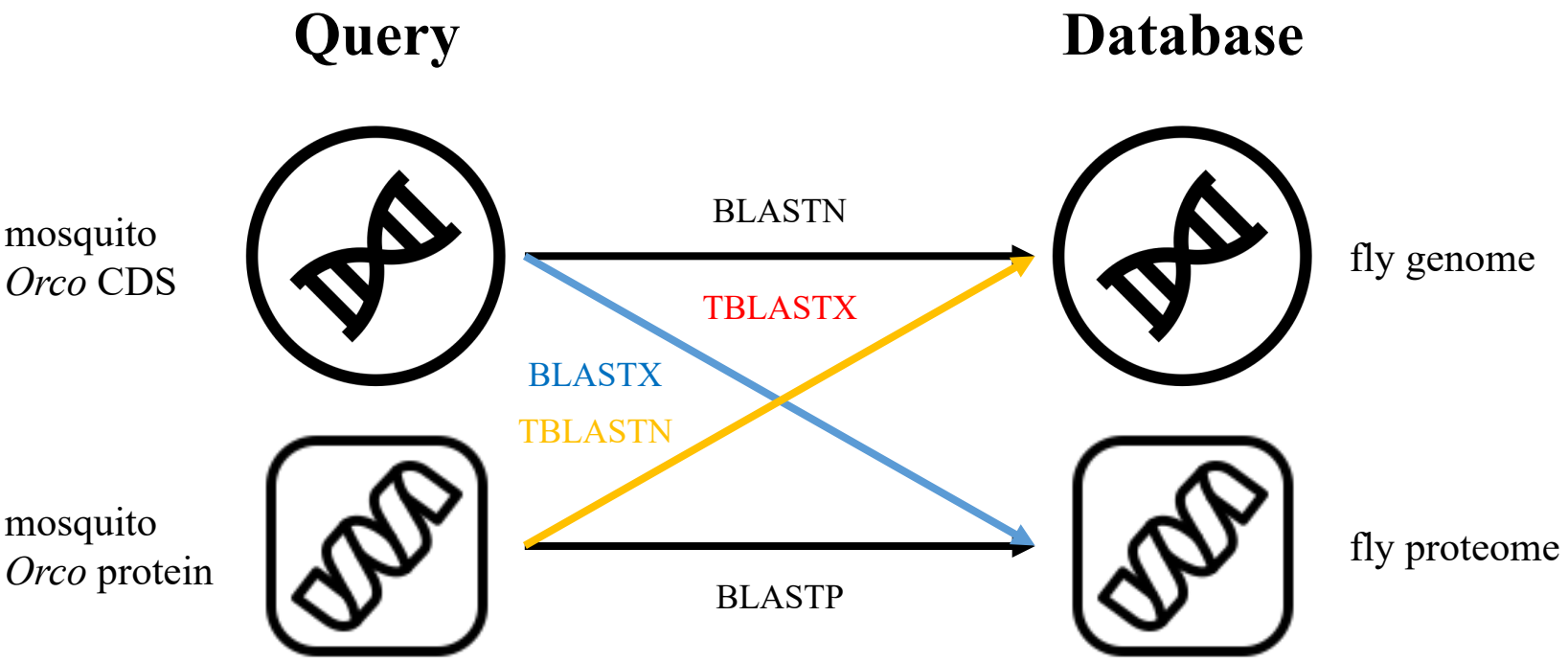


BLAST

- **B**asic **L**ocal **A**lignment **S**earching **T**ool
  - "*(it) finds regions of similarity between biological sequences. The program compares nucleotide or protein sequences to sequence databases and calculates the statistical significance*".

- Why local BLAST?
  - Poor internet connection
  - Too many sequences to analyze
  - Customized database and parameters

BLAST programs



Step 0: get help

- "[*command*] -h" (for a list of options)
  "[*command*] -help" (for detailed help information)

- available programs:
  - blastn
  - blastp
  - tblastn
  - blastx
  - tblastx
  - deltablast
  - psiblast
  - makeblastdb
  - blastdbcmd

Step 1: prepare inputs

- **FASTA** format:
  - each sequence is represented by:
    1. one line of sequence id (and description);
       the line has to start with “>” !!!
    2. followed by one or more lines of sequence data;

- Example:

```
line 1: >sp|Q6GZX4|001R_FRG3G Putative transcription factor 001R
        OS=Frog virus 3 (isolate Goorha) OX=654924 GN=FV3-001R PE=4
        SV=1
line 2: MAFSAEDVLKEYDRRRRMEALLLSLYYPNDRKLLDYKEWSPPRVQVECPKAPVEWNNPPS
line 3: EKGLIVGHFSGIKYKGEKAQASEVDVNKMCCWVSKFKDAMRRYQGIQTCKIPGKVLSDLD
line 4: AKIKAYNLTVEGVEGFVRYSRVTKQHVAAFLKELRHSKQYENVNLIHYILTDKRVDIQHL
line 5: EKDLVKDFKALVESAHRMRQGHMINVKYILYQLLKKHGHGPDGPDILTVKTGSKGVLYDD
line 6: SFRKIYTDLGWKFTPL
```

Step 2: make BLAST database

- Protein database:

  makeblastdb -dbtype prot -in fly.proteins.fasta \
  -out fly.proteins -parse_seqids

- Nucleotide database:

  makeblastdb -dbtype nucl -in fly.genome.fasta \
  -out fly.genome -parse_seqids

- Prebuilt BLAST databases available from NCBI:
  - nr, nt, refseq_protein, refseq_genomic…
  - e.g., update_blastdb nr

Step 3: run BLAST

- BLASTP:
  blastp -db fly.proteins -query mosquito.pep -out mosquito.blastp.out

- TBLASTN:
  tblastn -db fly.genome -query mosquito.pep -out mosquito.tblastn.out

- BLASTX:
  blastx -db fly.proteins -query mosquito.cds -out mosquito.blastx.out

- BLASTN:
  blastn -db fly.genome -query mosquito.cds -out mosquito.blastn.out1
  blastn -task blastn -db fly.genome -query mosquito.cds \
  -out mosquito.blastn.out2

Step 3: run BLAST

- DELTA-BLAST

  deltablast -db fly.proteins -rpsdb mini_deltablast \
  -query mosquito.pep -out mosquito.deltablast.out \
  -show_domain_hits

- Other important options:

  -evalue: *E*-value cutoff (e.g., 1e-5)
  -num_threads: number of threads (e.g., 2)
  -outfmt: output format (e.g., 6)
  for "-outfmt 0" (default):
  -num_alignments: max number of hits to show alignments (e.g., 5)
  -num_descriptions: max number of hits to show descriptions (e.g., 5)
  for "-outfmt 6":
  -max_target_seqs: max number of hits to report (e.g., 5)

Step 4: check output

- Check your BLAST output with “more” (or “less”)
- Compare the results of BLASTP and DELTA-BLAST, what is the difference?
- Run your BLAST analysis with and without the “-outfmt 6” option (*remember to modify the output file name*), what is the difference?
- Run your BLAST analysis with different e-value cutoffs, what is the difference?

Step 5: extract selected sequences from database

- Get one sequence:

```
blastdbcmd -db fly.proteins -entry FBpp0070000 \
    -out FBpp0070000.pep
```

- Get a number of sequences:

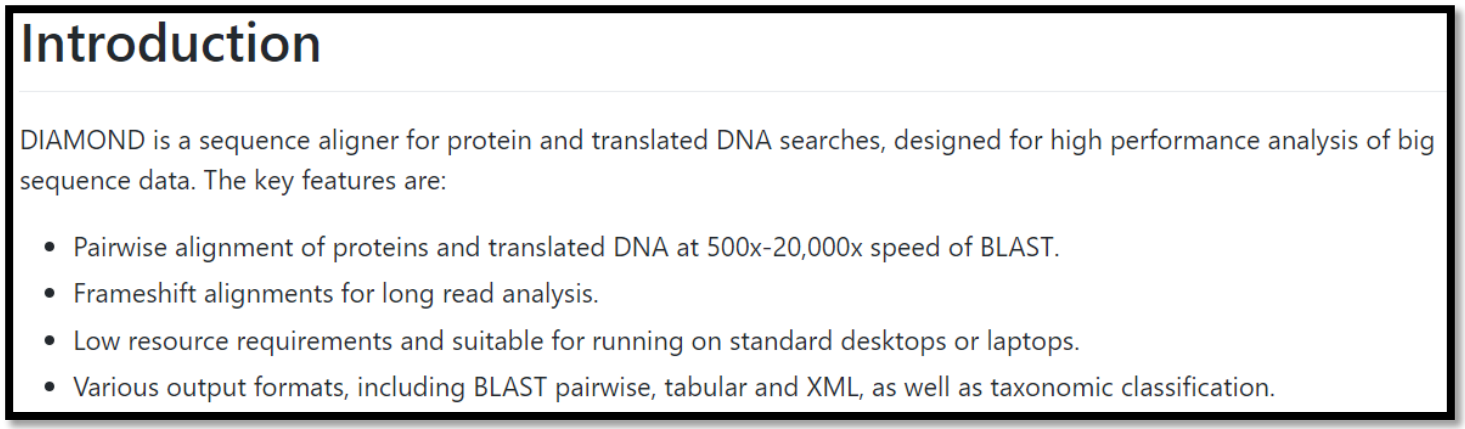
```
blastdbcmd -db fly.proteins -entry_batch selected.list \
    -out selected.pep
```

- Get all sequences:

```
blastdbcmd -db fly.proteins -entry all -out all.fa
```

Is BLAST quick enough for you?

- Alternatives:
  - GPU-supported BLAST
  - USEARCH
  - DIAMOND
  - …



Introduction

DIAMOND is a sequence aligner for protein and translated DNA searches, designed for high performance analysis of big sequence data. The key features are:

- Pairwise alignment of proteins and translated DNA at 500x-20,000x speed of BLAST.
- Frameshift alignments for long read analysis.
- Low resource requirements and suitable for running on standard desktops or laptops.
- Various output formats, including BLAST pairwise, tabular and XML, as well as taxonomic classification.

DIAMOND try-on

- Get help:
  diamond help

- Make database:
  diamond makedb -d fly.proteins --in fly.proteins.fasta

- "BLASTP" search:
  diamond blastp -d fly.proteins -q fly.proteins -o fly.proteins.diamond.out

- Other important options:
  -e: *E*-value cutoff (e.g., 1e-5)
  -p: number of threads (e.g., 2)
  -f: output format (e.g., 6)
  -k: max number of hits to report (e.g., 5)
  --sensitive or --more-sensitive: more sensitive search

Questions?